1972

An implicit enumeration algorithm for solving the multi-period warehouse location-allocation problem

William A. Engel
Lehigh University

AN IMPLICIT ENUMERATION ALGORITHM
FOR SOLVING THE MULTI-PERIOD
WAREHOUSE LOCATION-ALLOCATION PROBLEM

by

William A. Engel, Jr.

A Thesis

Presented to the Graduate Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in

Industrial Engineering

Lehigh University

1972

CERTIFICATE OF APPROVAL

This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science.

May 9, 1972

Date

Gary E. Whitehouse

Professor in Charge

Chairman of the Department of Industrial Engineering

ACKNOWLEDGMENTS

The author is grateful to his advisors, Dr. Gary E. Whitehouse of Lehigh University and Mr. Anthony J. Anastation of Western Electric Engineering Research Center, for their assistance in developing the topic and in preparing this thesis.

The author also wishes to acknowledge the excellent skill and cooperation of Miss Donna Sutphin in typing this thesis.

The author is especially grateful to his family for their patience and understanding while the thesis was being prepared.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Various types of location-allocation problems are introduced with reference to past efforts to solve these problems. An implicit enumeration algorithm is developed to solve the location-allocation problem of determining the optimum size and location of warehouses in a distribution system consisting of factories, warehouses, and demand areas. Warehouse and factory capacity constraints are present. A heuristic-like approach is used to determine an initial feasible solution and a branch and bound technique is used to search for the optimum solution. An example is followed step-by-step to show the algorithm's operation.

Computational aspects of the problem are discussed and sensitivity to cost and demand changes are investigated. The algorithm was found to find the optimum solution for moderate size problems in a reasonable amount of computer time. Large problems required a large, but not prohibitive, amount of computer time. Computer memory was not a limiting factor. The algorithm was found to be relatively insensitive to cost changes and only moderately sensitive to demand changes.

I. INTRODUCTION

The process of location-allocation involves determining the location of facilities such as factories or warehouses, to supply a set of destinations and determining how these facilities will satisfy the demand at these destinations.

This area of study has come into prominence only in the last decade. The field was apparently opened in 1963 by Cooper [3] who defined the location-allocation problem and discussed its computational aspects. Cooper stated the general location-allocation problem as follows:

"Given: (1) The location of each destination.
(2) The requirements at each destination
(3) A set of shipping costs for the region of interest.

To Determine:
(1) The number of sources.
(2) The location of each source.
(3) The capacity of each source."

There are many types of location-allocation problems. Among them are the problems of:

(1) Determining the location of warehouses with respect to demand.
(2) Locating plants with respect to demand.
(3) Locating warehouses with respect to plants and demand.

In some cases the problems are also considered over a planning horizon in which cases the added dimension of time is introduced.

If all costs were linear these types of location-allocation problems could theoretically be solved with linear programming. Transportation costs are generally close enough to being linear so that the linearity assumption can be made. However, the costs of warehouse or plant operations are highly non-linear since there is a large initial cost of constructing the building and there are many operational costs which are independent of volume. A good approximation to the non-linear cost can be made using the concept of fixed charge plus linear cost. This concept assumes that when a facility is opened a cost is encountered which is independent of the volume that is handled by the facility and that all other costs are a linear function of the volume.

For purposes of computation the idea of grouping all demand in a certain area into a central point has become popular. Instead of considering the location of every unit of demand, we now consider only a reasonable number of locations with each location representing the demand in its vicinity. These locations are called "demand centers."

The problem which is the subject of this paper is that of determining the optimal size and location of warehouses in a distribution system over a series of time periods. The distribution system consists of a predetermined set of factories, a predetermined set of demand centers, and the warehouses. The warehouse locations are picked from a set of potential locations. Each potential location has one or more possible increments of capacity from which to choose. If a warehouse site is in the solution for any period is must remain in the solution for all subsequent periods with no decrease in capacity. The costs to be minimized are the transportation costs from factory to warehouse to demand center and the warehouse costs. Transportation costs are assumed to be linear with volume and warehouse costs are assumed to be fixed charge plus linear.

In optimizing over a time horizon of several periods, the decisions for a given period are dependent on the decisions for other periods. In effecting this "linkage" linear programming models tend to become very large and cumbersome. The problem defined above can be solved using zero-one integer linear programming. The formulation for this problem is given in Appendix A. As can be seen the number of variables and the number of constraint equations and inequalities become very large for moderate size problems. For example, a problem with one factory, five potential warehouse locations with three possible capacities each, 25 demand centers, and ten time periods would require 3900 zero-one variables and 4660 constraint equations or inequalities.

This problem can also be solved by considering many smaller problems, optimizing each of the smaller problems, choosing various combinations of the solutions to the smaller problems in such a way as to maintain the feasibility of the solutions, and finding which combination has the lowest cost. This approach to the problem is developed into an implicit enumeration algorithm in Chapter III for solving the particular location-allocation problem defined above.

Chapter II presents a summary of various approaches to this and other variations of the location-allocation problem found in the literature. Chapter III contains the algorithm for optimally solving this problem. Chapter IV discusses the results of this algorithm and Chapter V presents the conclusions to be drawn. Chapter VI suggests areas for further study.

Appendix A gives the zero-one integer linear programming formulation to the problem stated above for the purpose of having an exact statement of problem. Appendices B and C provide background material for the algorithm, Appendix B being the justification for using linear programming for solving the two-stage transportation problem (factory to warehouse to demand center) and Appendix C being a brief introduction to the branch and bound technique.

II. SURVEY OF THE LITERATURE

As mentioned in Chapter I it is apparently Cooper who is credited with spurring the recent interest in the location-allocation problem. However, Cooper cites references as far back as 1647 in which the problem was considered. Cooper presents several methods for finding the optimal location of sources and discusses the computational aspects. Cooper's work, however, locates sources on a continuous plane, whereas most work since that time has been in choosing from a set of potential source locations.

Since Cooper's work many location-allocation algorithms have appeared. Some use a heuristic solution procedure in which "rules of thumb" are applied to give a good, but not necessarily optimal solution. Other algorithms have been developed to give the exact or nearly exact solution.

Kuehn and Hamburger [9] developed a heuristic solution to the problem of locating warehouses in a single time period. Their solution involves starting with one warehouse and adding one warehouse at a time until no further decrease in total cost is possible. The algorithm makes an attempt to replace non-economic warehouses with lower cost warehouses. Such replacements are made until no further cost reduction is possible. A "good" solution is obtained. Demand areas are assigned to warehouses by finding the minimum sum of shipment costs and warehouse expansion costs. Facility costs are assumed to be of the fixed charge plus unit cost type.

Feldman, Lehrer, and Ray [7] have developed a similar heuristic algorithm. They assume a continuous concave cost function and begin with all warehouses in solution and drop one warehouse at a time until no further reduction in cost results. In this manner a good solution is obtained.

Dachel [4] has developed a heuristic for solving the multi-product warehouse location problem. His approach is similar to that of Kuehn-Hamburger, however, he uses a three index transshipment model to obtain minimum cost for a given set of warehouses.

Effroymson and Ray [5] solve the uncapacitated problem by formulation as an integer programming problem. The problem is then solved with branch and bound. Plant costs are assumed to be fixed. Extensions are given for fixed and variable plant costs, and for using linear segments to approximate concave plant costs.

Spielberg [14] solves the uncapacitated plant location problem with a branch and bound algorithm applying a linear program at each stage. Side conditions, such as a minimum or maximum on the number of plants open, are allowed in the algorithm. Extensions are considered for the capacitated location problem and the fixed charge problem.

Baumol and Wolfe [2] use the linear programming transportation model in an iterative procedure to solve the location-allocation problem. Non-linear cost functions are accounted for by using the marginal cost at each iteration. An approximately optimal solution is obtained.

Sa [13] uses an out-of-kilter algorithm as part of a branch and bound procedure to solve the plant location problem with capacity constraints. He also presents an approximate solution using a heuristic.

Marks [11] uses a branch and bound algorithm to solve the problem of locating warehouses giving consideration to both factories and demand centers. He uses an out-of-kilter algorithm at each stage.

Revelle, Marks, and Liebman [12] compare various models and discuss the solution techniques of six of these.

Ellwein and Gray [6] tabularized the characteristics of 15 location-allocation models in terms of what problems they would solve and whether or not they provided an exact solution. A branch and bound algorithm is developed for solving a general location-allocation problem using network flow techniques to solve the transportation problem at each iteration.

Of the above models only Marks considers the warehouse location problem in the context of a warehouse being an intermediate point between factory and consumer. None of the above models consider optimizing over a time horizon of several periods.

Anastation and Weingartner [1] have developed a location-allocation model which produces warehouse strategies over a planning horizon. A heuristic algorithm, similar to that of Kuehn and Hamburger, is applied for each time period and for different demand estimates within that period. A number of alternate configurations are generated for each period. The results for all periods are then considered with weighting factors and a score for each location is determined. The location with the highest score is considered for opening in the first period. The same procedure is repeated with the first period solution being an existing configuration at the second period. This procedure continues until the end of the planning horizon is reached.

The algorithm developed in Chapter III of this paper solves the same form of the location-allocation problem as solved by Anastation and Weingartner. The Anastation-Weingartner algorithm considers demand as being uncertain, but with a range of expected values, and finds a solution which will be good at all anticipated demand levels. The algorithm in this paper considers demand as being deterministic and finds the minimum cost solution.

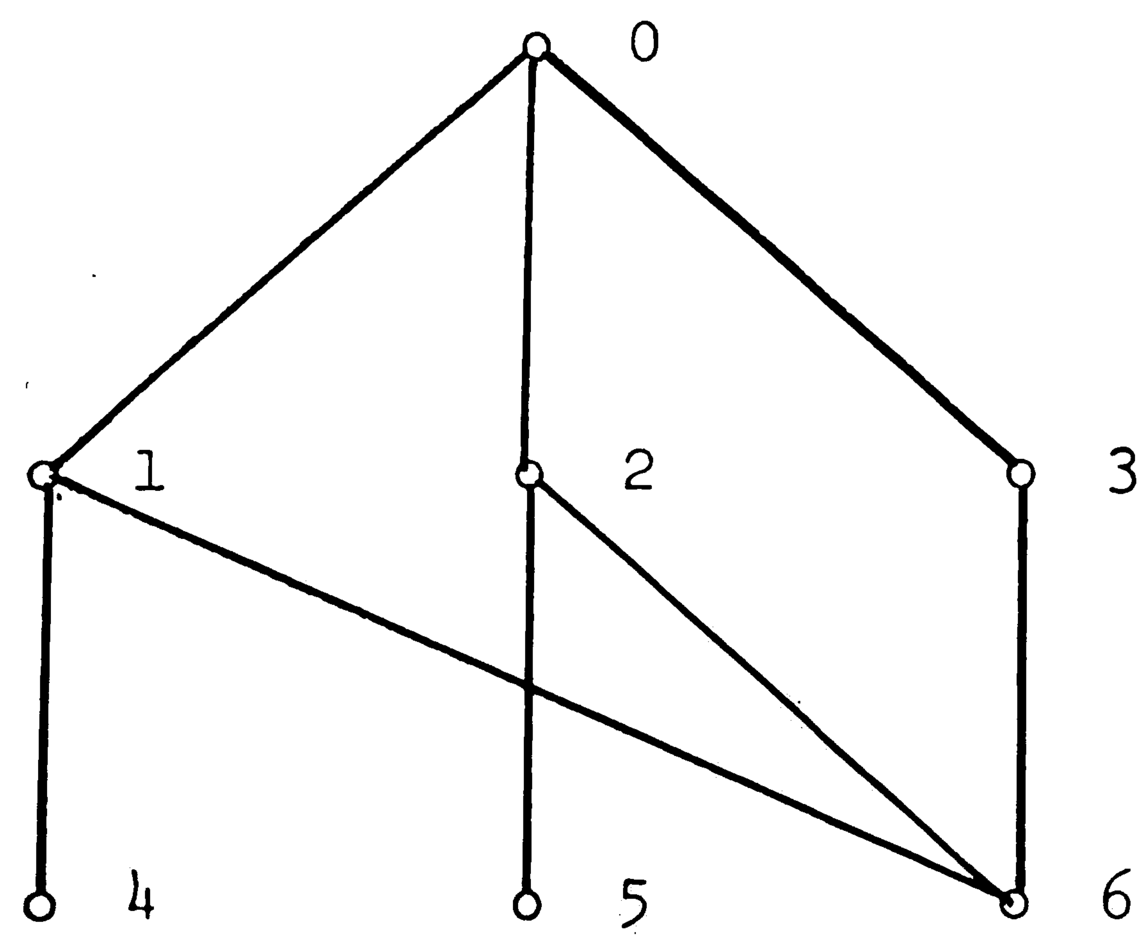
III. THE ALGORITHM

This chapter will develop an implicit enumeration algorithm to solve the warehouse location-allocation problem described in Chapter I and for which a zero-one integer linear programming formulation is given in Appendix A.

The algorithm solves this problem by considering many smaller problems. A branch and bound procedure is used to eliminate infeasible and/or non-optimal solutions. Refer to Appendix C for a description of branch and bound. The search tree for this algorithm consists of nodes and branches. Each node represents a problem in which the time period is fixed and a particular configuration of warehouses and warehouse capacities is also fixed. All nodes for a given time period are on the same level of the tree. The source node is level zero, level one nodes are those for the first period, level two for the second period, etc. Each branch is a bridge from one period to the next. The starting node for the branch is the configuration in a given period and the terminal node is the configuration in the following period. Associated with each node is the cumulative cost of the system through the current period.

Suppose we wish to construct a search tree for the case of two potential sites and two time periods. Figure III-1 shows what such a search tree would look like. The three branches emanating from the source node represents, in the first period, the opening of warehouse one (branch [0,1]), warehouse two (0,2), and both warehouses (0,3),

EXAMPLE OF A SEARCH TREE



Node	Period	Warehouses Open
0	Source Node	
1	1	1
2	1	2
3	1	1,2
4	2	1
5	2	2
6	2	1,2

Figure III-1

respectively. The branch (1,4) indicates that warehouse one was open in period one and is the only warehouse open in period two. Branch (1,6) shows the addition of warehouse two in period two. Branch (2,5) indicates warehouse two is open by itself in period two and (2,6) shows the addition of warehouse one in period two. Branch (3,6) shows both warehouses remaining open through both periods. Branches which would result in the closing of a warehouse are not allowed since an assumption has been made that once a warehouse has been opened it will remain open. The branches not possible are (1,5), (2,4), (3,4), and (3,5).

At each node of the search tree it is necessary to determine the cost of operating the system with the given configuration during the given period. Since the node itself determines which warehouses will be open and which period is being considered the fixed costs are predetermined, that is the optimization of costs at that node does not involve fixed costs. The problem to be solved at the node is to minimize the sum of the transportation cost from factory to warehouse, the handling cost at the warehouse, and the transportation cost from warehouse to demand center. Appendix B shows that, using the transshipment concept, this problem can be solved as an ordinary linear programming transportation problem. Therefore, the variable costs can be optimized for each node of the search tree. This variable cost plus the fixed costs constitute the cost of operating the system for the time period.

As part of the algorithm the cumulative cost must be known at each node. For nodes with only one predecessor (such as 4 and 5 of the example) this is simply the sum of the cost of the current node and the cumulative cost through the previous node. For nodes with more than one predecessor the predecessor with the lowest cumulative cost is the one chosen to determine the cumulative cost of the current node.

Thus far, only the "branch" portion of branch and bound has been considered. If one had to analyze every node, even modest size problems would be computationally infeasible. Therefore, the "bound" concept must be used to limit consideration only to those nodes which are feasible and could possibly lead to an optimum solution.

Another assumption must be made at this point, that is that the system cost for any period must be at least as great as the cost for the previous period. This assumption is not unreasonable, especially for planning purposes since an increase in demand and/or cost is usually assumed. This assumption results in the cumulative cost versus time relationship shown by the curved line in Figure III-2. The straight line indicates the extreme case where the cost per period remains constant from period to period. If a trial optimum solution (one that is feasible but not known to be optimum) has been found, the straight line of Figure III-2 can be constructed. Each node of the search tree can be compared with the value of straight line at the period represented by the node. If the cumulative cost for the node lies above the line no branches need to be construc-

CUMULATIVE COST CURVE

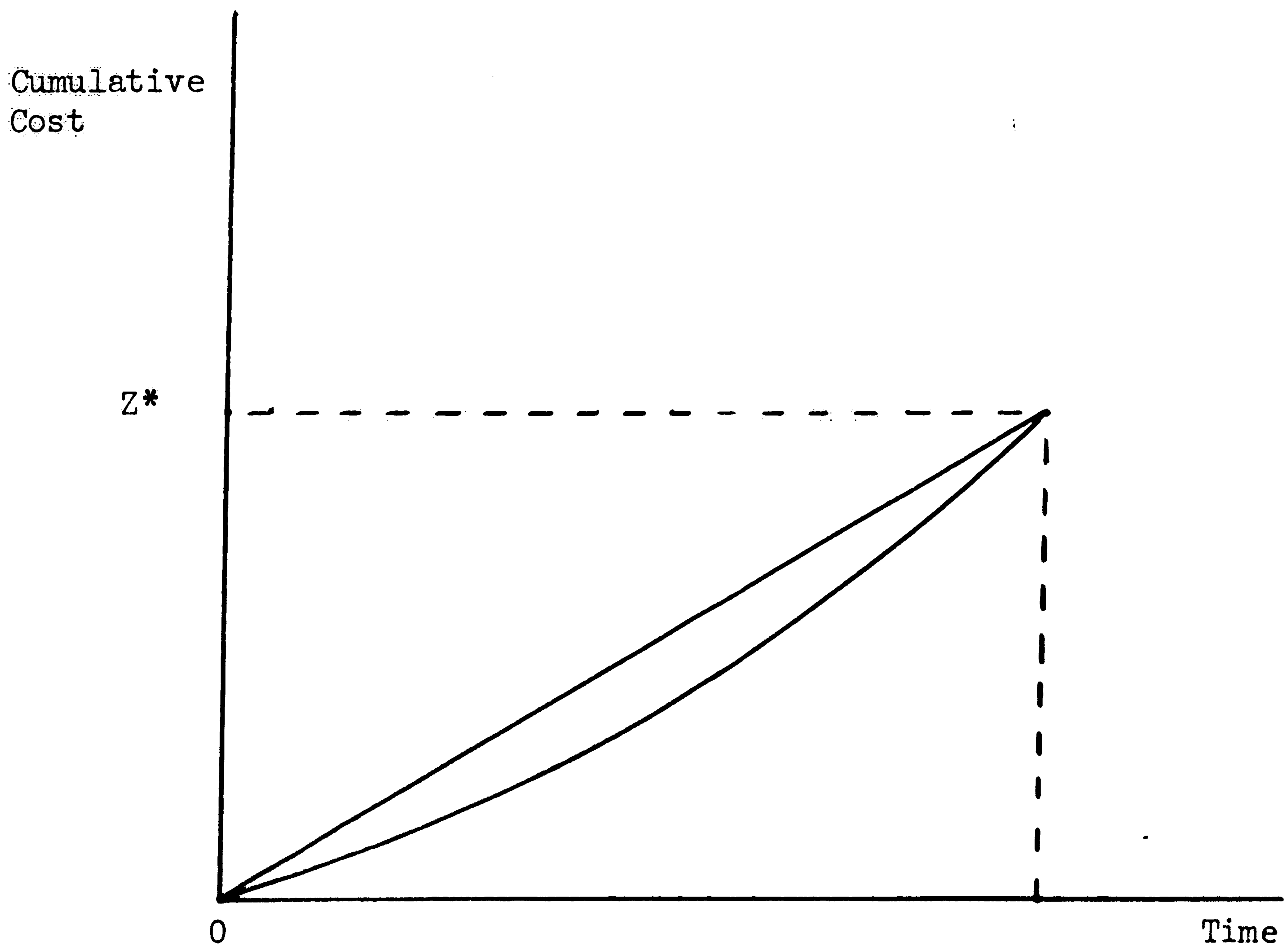


Figure III-2

ted from the node since the cumulative cost in succeeding periods could not fall below the line. If the cumulative cost is on or below the line branches must be constructed since the node could possibly lead to a lower cost solution than the trial optimum. If a new trial optimum is found the line is adjusted downward.

The concept of using this straight line as the criteria for removing nodes from consideration has introduced the bounding technique to be used. The upper bound for each period can now be defined in terms of the trial optimum solution. Let Z* be the cumulative cost through the last period of the trial optimum solution. Let N be the number of periods. Therefore, define the upper bound for the t-th period to be

$$UB_t = \frac{t}{N} \cdot Z^*$$

Once a value for Z* has been determined it is possible to apply an upper bound at each level of the search tree and reduce considerably the number of nodes that need to be considered.

The following paragraphs describe the implicit enumeration algorithm that has been developed by the author to solve the location-allocation problem where it is desired to optimally locate warehouses and determine their size over a planning horizon in a distribution system consisting of factories, warehouses, and demand centers.

There are two basic steps involved in the algorithm. The first is to determine an initial feasible solution. The cost of this solution will become the value of Z* to be initially used in the second step. The second step is to apply the branch and bound principle to the problem and find the optimal solution or prove the initial solution to be optimum.

The first step is accomplished by first considering all configurations of warehouses which will meet the demand in the first period using the smallest increment of capacity of each warehouse. (The smallest increment was chosen because it would limit the number of feasible configurations. However, the choice is arbitrary unless the combined capacity of all potential warehouses at their smallest capacity is less than the demand.) Figure III-3 shows the sequence for generating warehouse configurations.

Next, find the cost of operating each configuration in the first period ignoring warehouse and factory capacity constraints. This task involves only finding the minimum cost supply route for each demand center, rather than solving a time consuming transportation problem for each configuration. The configuration with the lowest cost is then chosen as the initial first period solution. The actual cost of this configuration is found by using the transportation algorithm.

This configuration is checked to see if it can handle the second period's demand. If it can the cost is determined for the unconstrained problem as was done in period one. If it cannot handle the demand the unconstrained problem is not solved. The configuration is now augmented by the addition of another warehouse at the first unoccupied potential site. The cost is again determined for the unconstrained problem. The configuration from period one is again augmented by the

SEQUENCE FOR GENERATING WAREHOUSE CONFIGURATIONS

1. One warehouse at a time:

   (i),i=1,2,3,...,N

   N combinations.

2. Two warehouses at a time:

   (i,j),j=1,2,3,...,N-1;

   i=j+1,j+2,...,N, for each j;

   N(N-1)/2 combinations.

3. Three warehouses at a time:

   (i,j,k),k=1,2,3,...,N-2

   j=k+1,k+2,...,N-1, for each k;

   i=j+1,j+2,...,N, for each j;

   N(N-1)(N-2)/6 combinations.

.
.
.

N-1. N-1 warehouses at a time:

   (i,j,k,...,m),m=1,2;

   ...

   j=k+1,k+2, for each k;

   i=j+1,j+2, for each j;

   N combinations.

N. N warehouses at a time:

   (N,N-1,N-2,...,3,2,1);

   1 combination.

Total combinations = 2 - 1.

Figure III-3

next unoccupied site and the cost determined without capacity constraints. When all unoccupied sites have been considered the configurations are examined to determine the one with the lowest cost. The actual cost of this configuration is found by the transportation algorithm. This configuration is now the second period's initial solution and may be identical to the first period's or have one additional warehouse.

The third period's initial solution is found by considering the configuration of the second period's initial solution and following the same procedure as was done in period two. Subsequent periods are handled in the same manner. When the initial solution for the last period is found, the cumulative cost of the initial solutions for all periods becomes Z*. Initial upper bounds are now set for all periods using the formula above.

Now the second step can begin. The first warehouse configuration is generated. Capacities of the warehouses in the configuration are set to their smallest increment. This configuration is then checked to determine if it can meet the demand requirements for the first period. If it cannot then go to the next combination of capacity increments and try again. When a configuration which meets the capacity constraints is found, the fixed warehouse costs are calculated. If these costs exceed the upper bound for period one then this configuration of warehouses and capacities is dropped. If this is the smallest increment of capacity for each warehouse in the dropped configuration go on to the next configuration since addition of capacity cannot lower fixed costs. Otherwise go on to the next combination of warehouse capacities.

The configurations which have passed the first bounding test are now subjected to a second test. The unconstrained allocation problem is solved as was done in the first step. If the cost of a configuration (including fixed costs) from the unconstrained problem is greater than the upper bound the configuration can be dropped since the cost of the constrained problem is at least as great as the cost of the unconstrained problem. If all warehouses in the dropped configuration were at their lowest capacity increment go on to the next configuration. If not go on to the next combination of capacity increments.

A final test is now applied to those configurations which have passed the second test. The constrained problem is solved by applying the transportation algorithm. The cost obtained plus the fixed costs of the configuration are compared to the upper bound for the first period. If the cost is less than or equal to the upper bound, the cost, configuration, and capacities are saved to pass on to the next period. Also at this point if no cost reduction is obtained by adding capacity to a warehouse, then only the lower capacity solution is saved. Now the next combination of capacity increments is tested against demand and the step is repeated. If a transportation problem was solved with all warehouses at their smallest capacity, reference to this solution will show which warehouses were operating at their full capacity (no slack capacity). Only these warehouses need to be considered for combinations of capacity increments.

When all configurations have been considered a set of feasible solutions with cost less than or equal to the upper bound has been saved for the first period. The branches of the search tree have now been constructed from the source node to the first level nodes.

The lowest cost configuration and capacity combination for the first period is chosen from those saved. This configuration is checked to see if it has sufficient capacity to meet the demand. If it cannot meet the demand it must be augmented first. If it can meet the demand, the fixed costs are determined and checked against the upper bound. If the fixed costs plus the cost of the configuration in period one is greater than the upper bound for period two, the configuration cannot lead to an optimal solution; return to period one and use the next best configuration. If the total cost is less than or equal to the upper bound, the unconstrained problem is then solved and its costs (including fixed costs) plus the cost of the configuration in the first period is compared to the upper bound for the second period. If the total cost exceeds the bound the configuration must be augmented. If the total cost is less than or equal to the bound, the constrained problem is then solved using the transportation algorithm. The cost from the transportation problem, plus the fixed costs, plus the cost from period one is compared to the upper bound. If less than or equal to the bound the solution is saved for use in period three. If greater than the bound, the next capacity combination is considered for those warehouses which are at capacity.

After all combinations of capacity have been considered, the configuration is augmented by adding a warehouse at an unoccupied potential site and repeating the above steps for each of the unoccupied sites. Those solutions saved are either identical to the first period lowest cost solution or have one additional warehouse. Branches have now been constructed from one first level node to second level nodes.

The lowest cumulative cost configuration and capacity combination for the second period is chosen. This configuration is used to construct third period solutions in the same manner as the second period solutions were constructed.

This process is continued until the final period is reached. The lowest cumulative cost through the final period becomes the currently optimum solution. The configuration and capacity combinations from every period are saved as the optimum solution.

The process of backtracking up the search tree begins. The next best configuration from the period preceding the final period is chosen and solutions to the final period are constructed on it. This continues until there are no more solutions remaining at the period preceding the final period. The algorithm backtracks another period, picks the next best, and works toward the final period.

The backtracking continues until there are no more solutions left in the first period. Each time the final period is reached the solution is checked against the currently optimum solution. If it is better it becomes the new currently optimum solution. The solution which is currently optimum when the search terminates is the optimum solution.

The algorithm has been programmed in FORTRAN on the PDP-10 computer. Copies are available from the author. To help clarify the algorithm, Figure III-4 presents the algorithm step-by-step.

As an example consider a distribution system with one factory, four demand centers, two potential warehouse sites with one increment of capacity each and two time periods. Figure III-5 shows the costs, demands, and capacities for the system.

First, a warehouse is considered for site one in period one. Total demand to be handled in period one is 470 units. The capacity at site one is 600, therefore, this warehouse is feasible. Now the unconstrained problem is solved and the costs are found to be:

Warehouse Fixed			1000
Factory to Warehouse	$0\cdot470$	=	0
Warehouse to Demand Center	$0\cdot100+25\cdot150+20\cdot120+2\cdot100$	=	8150
Warehouse Variable			0
Total			9150

Warehouse configuration (1) has an unconstrained cost of 9150 for period one.

Now a warehouse is considered for site two. Since its capacity is 600 it is also feasible. Its fixed cost is 1000 which is less than

STEPS OF THE ALGORITHM

(a) Period = 1, Min = Inf.

(b) Generate a warehouse configuration.

(c) Set capacities to first increment.

(d) Will this configuration meet demand?

Yes (e)
No (h)

(e) Is fixed cost less than Min?

Yes (f)
No (h)

(f) Is minimum cost to supply each demand center without capacity constraints plus fixed cost less than Min?

Yes (g)
No (h)

(g) Let Min = cost from (f).

(h) Have all configurations been considered?

Yes (j)
No (i)

(i) Generate next configuration and go to (c).

(j) Find optimal transportation cost for configuration with minimum cost determined above.

(k) Period = 2, Min = Inf.

(l) Use previous period's configuration.

(m) Will this configuration meet this period's demand?

Yes (n)
No (q)

(n) Is fixed cost for this configuration plus cost accumulated through the previous period less than Min?

Yes (o)
No (q)

Figure III-4

(o) Is minimum cost to supply each demand center without capacity constraints, plus fixed costs, plus cost accumulated through the previous period less than Min?

Yes (p)
No (q)

(p) Let Min = cost from (o)

(q) Consider previous period configuration with the addition of of one warehouse at a potential location. Repeat steps (m) through (q) for each potential location.

(r) Find optimal transportation cost for configuration with minimum cost as determined above.

(s) Have all periods been considered?

Yes (u)
No (t)

(t) Increment Period, Min = Inf. Go to (l)

(u) Set upper bound for each period.

(v) Period = 1.

(w) Generate a warehouse configuration.

(x) Set capacities to first increment.

(y) Will this configuration meet demand?

Yes (z)
No (ad)

(z) Is fixed cost less than upper bound for period 1?

Yes (aa)
No (ai)

(aa) Is minimum cost to supply each demand center without capacity constraints plus fixed cost less than upper bound?

Yes (ab)
No (ad)

Figure III-4 (con't.)

(ab) Is optimal cost to supply each demand center with capacity constraints plus fixed cost less than upper bound?

Yes (ac)
No (ad)

(ac) Save this configuration and capacity combination for future reference. Go to (ae).

(ad) Drop this configuration and capacity combination.

(ae) Have all increments of capacity for this configuration been considered?

Yes (ag)
No (af)

(af) Choose next combination of capacity increments. Go to (y)

(ag) Have all possible configurations been considered?

Yes (ah)
No (w)

(ah) Period = 2.

(ai) Pick lowest cost configuration and capacity combination from the previous period.

(aj) Have this configuration and capacity combination already been considered for this period?

Yes (ak)
No (al)

(ak) Is the accumulated cost coming into this period less than the accumulated cost coming into the previous consideration of this configuration?

Yes (al)
No (aq)

Figure III-4 (con't.)

(al) Will this capacity combination meet this period's demand?

Yes (am)
No (aq)

(am) Is fixed cost for this configuration and capacity combination plus cost accumulated through the previous period less than the upper bound for this period?

Yes (an)
No (aq)

(an) Is minimum cost to supply each demand center without capacity constraints, plus fixed cost, plus cost accumulated through the previous period less than the upper bound for this period?

Yes (ao)
No (aq)

(ao) Is optimal cost to supply each demand center with capacity constraints, plus fixed cost, plus accumulated cost through the previous period less than the upper bound for this period?

Yes (ap)
No (aq)

(ap) Save this configuration and capacity combination for future reference. Go to (ar).

(aq) Drop this configuration and capacity combination for this period.

(ar) Have all increments of capacity for this configuration been considered?

Yes (at)
No (as)

(as) Choose next combination of capacity increments. Go to (al).

(at) Consider last period configuration with the addition of one warehouse at a potential location. Repeat steps (aj) through (as) for each potential location.

Figure III-4 (con't.)

(au) Were any configurations saved for this period?

Yes (av)
No (bc)

(av) Is this the final period?

Yes (ay)
No (aw)

(aw) Pick the configuration and capacity combination from this period with the lowest accumulated cost.

(ax) Increment period. Go to (aj).

(ay) Pick the configuration and capacity combination from this period with the lowest accumulated cost. .

(az) Is this cost less than the upper bound for this period?

Yes (ba)
No (bb)

(ba) This configuration and the configurations from pervious periods which lead to it are saved as the currently optimal solution. Upper bounds for each period are recalculated based on the new optimum.

(bb) Delete all configurations and capacity increments saved for this period.

(bc) Decrement Period. Is this period zero?

Yes (bf)
No (bd)

(bd) Delete configuration and capacity combination which was the last to be used in this period.

(be) Are there any configurations left for this period?

Yes (aw)
No (bc)

(bf) Optimum solution has been proven to be optimum. Stop.

Figure III-4 (con't.)

TWO WAREHOUSE EXAMPLE

```
D1 o o F1                  o D4
W1   o




D3 o                W2 o o D2
```

Shipping Costs ----------------------

From	To W1	To W2
F1	0	10

From	To D1	To D2	To D3	To D4
W1	0	25	20	20
W2	25	0	20	20

Demand ------------------------------

	D1	D2	D3	D4
Period 1	100	150	120	100
Period 2	110	160	130	110

Other costs and capacities -----------

All fixed costs are 1000
All variable warehouse costs are 0
All warehouse capacities are 600 units
Factory capacity is 10000 units

Figure III-5

9150. The unconstrained problem is solved with costs:

Warehouse Fixed		1000
Factory to Warehouse	$10 \cdot 470 =$	4700
Warehouse to Demand Center	$25 \cdot 100 + 0 \cdot 150 + 20 \cdot 120 + 20 \cdot 100 =$	6900
Warehouse Variable		0
Total		12600

Warehouse configuration (2) has an unconstrained cost of 12600 for period one which is greater than the cost of 9150 for configuration (1). Configuration (1) remains the candidate for the initial solution.

The next configuration is (1,2) which has a capacity of 1200 and is feasible. Its fixed costs are less than 9150. The unconstrained problem is more complicated than before since there are alternate routings to each demand center. Since the problem is unconstrained it is necessary only to find the minimum cost route to each demand center from the factory. Costs are:

Warehouse Fixed		2000
Factory to Whse. to D.C. 1	$100 \cdot \min(0+0, 10+25) =$	0
Factory to Whse. to D.C. 2	$150 \cdot \min(0+25, 10+0) =$	1500
Factory to Whse. to D.C. 3	$120 \cdot \min(0+20, 10+20) =$	2400
Factory to Whse. to D.C. 4	$100 \cdot \min(0+20, 10+20) =$	2000
Warehouse Variable		0
Total		7900

The cost for configuration (1,2) is 7900 and less than the cost of 9150 for configuration (1). There are no more possible configurations to consider for period one. Therefore (1,2) is the lowest cost unconstrained configuration for period one.

The true cost of configuration (1,2) is found by solving the constrained problem using the linear programming transportation algorithm. This cost is found to be 7900.

Moving on to period two, configuration (1, 2) is tested against period two demand which is 510. Configuration (1,2) has capacity 1200 and is feasible. The unconstrained problem is solved as before and the cost for period two is 8400.

Since configuration (1,2) contains all warehouses no other warehouses can be added to augment the configuration. All feasible combinations for period two have been considered.

The true (constrained) cost of configuration (1,2) is found by the transportation algorithm to be 8400 for a cumulative total of 16300 over both periods.

Since period two is the last period the first portion of the algorithm is completed and the initial upper bounds to be used in the second part of the algorithm can be set.

$$UB_1 = \frac{1}{2} \quad 16300 = 8150$$

$$UB_2 = \frac{2}{2} \quad 16300 = 16300$$

The second part begins by considering warehouse configuration (1) for period one. Much of the second step will be a repeat of the first step since only one increment of capacity is used for each warehouse and since the problem is small in size. As before, configuration (1) meets capacity. Its fixed cost of 1000 is less than the upper bound of 8150. The cost of the unconstrained problem is 9150 which is greater than the upper bound. Therefore configuration (1) is dropped from consideration for period one before the transportation problem is solved.

It is readily observable that configuration (2) will also be dropped since its unconstrained cost is 12600, well above the upper bound of 8150.

Configuration (1,2) will yield a cost of 7900 for period one from both the unconstrained and constrained problems. Since 7900 is less than the upper bound of 8150 the configuration is saved as possibly being part of an optimum solution.

Now that all possible combinations of warehouse configurations have been considered, period two is begun. The lowest cost solution for period one is (1,2) with cost of 7900. This solution is used as the basis for period two. As before, the cost of this configuration for period two is 8400 for a cumulative cost of 16300. Since the cumulative cost is less than or equal to the upper bound configuration (1,2) is possibly part of the optimum solution and is saved.

The search tree for the preceding is shown in Figure III-6. Because of the assumption of warehouses open remaining open no other branches can be drawn from configuration (1,2) in the first period to any other node in the second period. It is now necessary to backtrack to the first period and pick the node with the next best cost and build on it for the second period. But there are no other first period solutions since the use of the upper bound has eliminated configurations (1) and (2) because they could not possibly lead to the optimum solution.

Since the first period has been reached by backtracking and there are no configurations left at the first period to be considered the search terminates. The solution saved as optimum has been verified to be optimum. Figure III-7 shows the computer printout from the algorithm giving details on the optimum solution.

SEARCH TREE FOR EXAMPLE

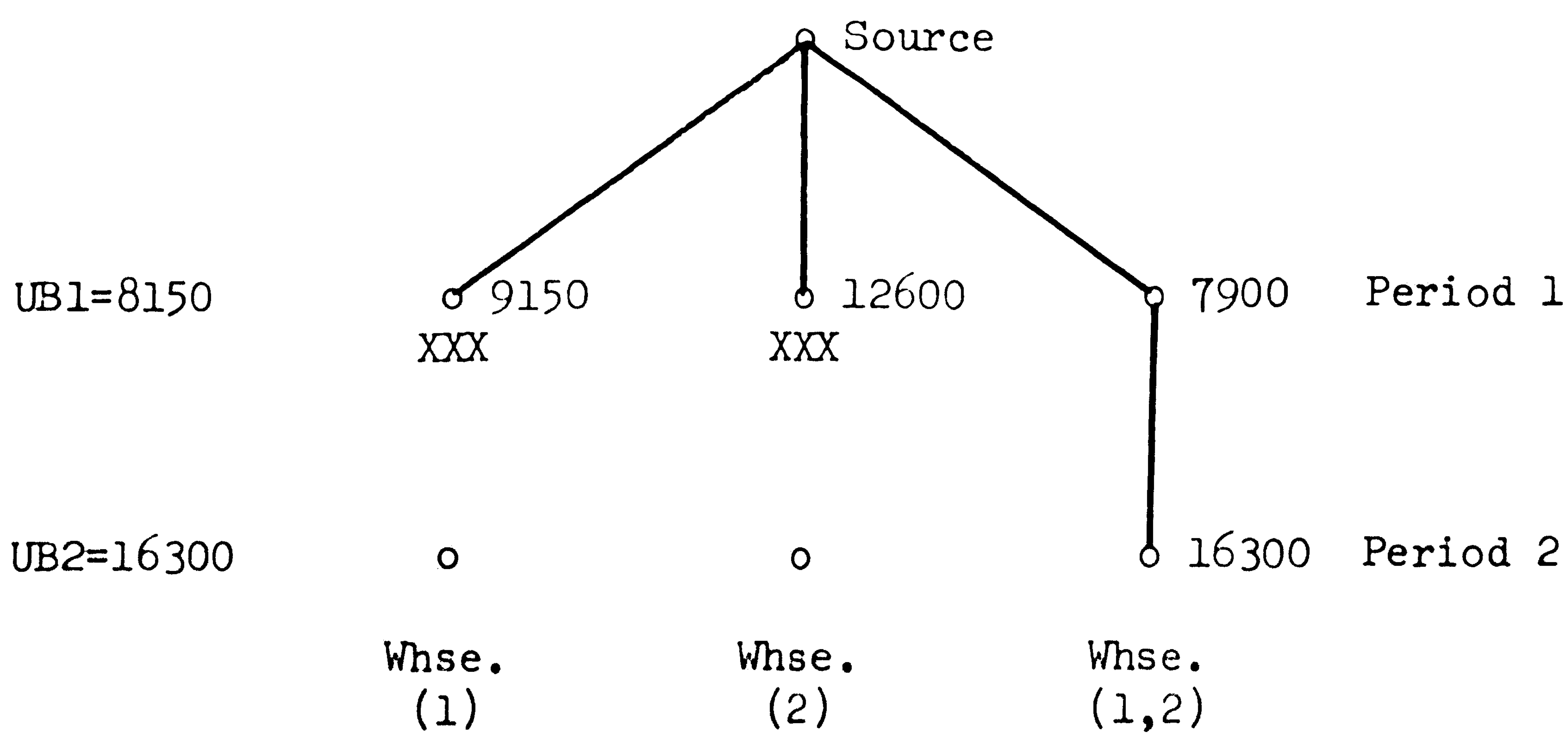


Figure III-6

COMPUTER OUTPUT FOR EXAMPLE

PERIOD 1

WAREHOUSES OPEN	2	1
CAPACITY INCREMENT	1	1

COST FOR PERIOD 1 $ 7900.00

CUMULATIVE COST THROUGH PERIOD 1 $ 7900.00

FACTORY TO WAREHOUSE MATRIX

	WHSE. 2	WHSE. 1
FACT. 1	150.0	320.0

TRANSPOSE OF WAREHOUSE TO DEMAND CENTER MATRIX

	WHSE. 2	WHSE. 1
D.C. 1	0.0	100.0
D.C. 2	150.0	0.0
D.C. 3	0.0	120.0
D.C. 4	0.0	100.0

PERIOD 2

WAREHOUSES OPEN	2	1
CAPACITY INCREMENT	1	1

COST FOR PERIOD 2 $ 8400.00

CUMULATIVE COST THROUGH PERIOD 2 $ 16300.00

FACTORY TO WAREHOUSE MATRIX

	WHSE. 2	WHSE. 1
FACT. 1	160.0	350.0

TRANSPOSE OF WAREHOUSE TO DEMAND CENTER MATRIX

	WHSE. 2	WHSE. 1
D.C. 1	0.0	110.0
D.C. 2	160.0	0.0
D.C. 3	0.0	130.0
D.C. 4	0.0	110.0

Figure III-7

IV. ANALYSIS OF THE ALGORITHM

The following criteria have been used to analyze the algorithm:

1. The maximum size problem which can be run with a given memory capacity.
2. The maximum size problem which can be run in a given amount of processor time.
3. The sensitivity of the algorithm to changes in input parameters.

The number of words of memory required by computer program can be determined from the following formula:

$$N = 9030 + 6i + 7j + 3k + 5t + 4ij + 3ik + 4jk + 3jl + kt + ijt + jkt + jlt$$

where:

N = Number of 36- bit words required (PDP-10)

i = Number of factories

j = Number of potential warehouse sites

k = Number of demand centers

l = Number of capacity increments at each site

t = Number of periods in the planning horizon

A problem with 5 factories, 10 potential sites with 3 capacity increments each, 50 demand centers, and 8 time periods requires 16,249 words of memory. The zero-one programming model of this problem requires 60,240 variables and 15,240 constraint equations or inequalities.

Since the algorithm is combinatorial in nature the execution time for a given size problem is not easily predictable. Table IV-I lists the processor time required for problems of various sizes. However, a general expression cannot be developed for the processor time because the time is highly dependent on factors in the input data such as the rate of increase in costs and demand. Also, the computer time required for an exhaustive experiment to determine the effect of various factors would be prohibitive.

The influence of the rate of increase of demand can be seen from the data of Table IV-2. This effect can be explained by referring back to Figure III-2. The area between the straight line and the curved line represents nodes of the search tree that need to be evaluated but do not lead to the optimum solution. As the rate of increase in demand gets larger, the area between the line and the curve gets larger and more nodes of the search tree need to be considered. As the rate increases all feasible nodes are being considered for some of the periods near the middle of the planning horizon. This contributes to the slowing of the increase in computing time seen at the higher rates in the table.

The effect on computing time of a change in the number of capacity increments is shown in Table IV-3. The test problem is one with one factory, five potential warehouse locations, 25 demand centers and eight periods. Based on this limited sample computing time is a linear function of the number of increments.

Each of the input costs and the demand were varied to determine the effect on the cost of the distribtuion system. First, the problem

EFFECT OF SIZE OF PROBLEM ON COMPUTING TIME

Number of Factories	Number of Potential Sites	Number of Demand Centers	Number of Periods	Number of Capacity Increments	Time to Optimum Solution (Seconds)	Time to End of Algorithm (Seconds)	Number of Transportation Problems
1	8	25	5	3	55.5	56.9	88
1	9	25	5	3	195.7	199.9	194
1	10	25	5	3	523.1	537.7	120
1	8	50	5	2	2393.8	2454.9	1442
1	10	50	5	2	3919.2	4150.5	1680

TABLE IV-1

EFFECT ON COMPUTING TIME OF FIXED PERCENTAGE DEMAND INCREASES

PROBLEM I

Percent Increase In Demand	Time to End of Algorithm (Sec.)	Number of Transportation Problems
0	3.4	16
1	6.2	43
2.5	8.9	62
5	18.4	125
7.5	39.5	271
10	67.1	413
12.5	82.3	487
15	88.2	533

PROBLEM II

Percent Increase In Demand	Time to End of Algorithm (Sec.)	Number of Transportation Problems
0	3.4	16
1	3.4	16
2.5	5.9	32
3	8.8	52
4	18.7	125
5	32.4	207
7.5	50.7	316
10	63.8	352

TABLE IV-2

EFFECT OF CAPACITY INCREMENTS ON COMPUTING TIME

Increments of Warehouse Capacity	Time to Optimum Solution (Sec.)	Time to End of Algorithm (Sec.)	Number of Transport. Problems
1	8.2	10.6	16
2	15.5	16.7	34
3	19.8	23.8	46

TABLE IV-3

was run with nominal input values and the allocation of demand centers to warehouses and warehouses to factories were determined for each period. Next all demand data were decreased ten percent, the algorithm was run again and the total cost obtained. This cost was compared to the cost using nominal demand.

Next the decreased demand was applied manually to the system and allocations generated by the nominal demand and the cost determined. This cost is the resultant cost of applying demand ten percent below nominal to a system designed for nominal. The cost was compared to the cost of a system designed for ten percent below nominal demand, the difference being the cost of the uncertainty in demand.

The above process was repeated for demand five percent below nominal, five percent above, and ten percent above.

Next the effect of fixed cost variations were determined by decreasing warehouse fixed costs by 20 percent for all warehouses and all periods, followed by a ten percent decrease, a ten percent increase and a 20 percent increase. The same steps were followed as for demand variations.

Also tested were factory to warehouse shipping costs, warehouse variable costs, and warehouse to demand center shipping costs. These parameters were tested at a ten percent decrease, five percent decrease, five percent increase, and ten percent increase.

The results of the sensitivity tests are given in Table IV-4. The third column gives the percent change in total cost from nominal, using the allocations determined by the nominal data, for each item

SENSITIVITY RESULTS

Parameter	Percent Change From Nominal	Percent Increase In Cost Due to Change	Percent Savings If Optimum Were Used
Demand	-10	-7.07	1.17
	- 5	-3.68	0.63
	5	4.17	0.39
	10	7.66	0.41
Fixed Costs	-20	-5.96	0.34
	-10	-2.98	0
	10	2.98	0
	20	5.96	0
Factory to Warehouse Shipping Costs	-10	-2.66	0.07
	- 5	-1.33	0.03
	5	1.33	0
	10	2.66	0
Variable Costs	-10	-0.99	0
	- 5	-0.50	0
	5	0.50	0
	10	0.99	0
Warehouse to Demand Center Shipping Costs	-10	-3.36	0
	- 5	-1.68	0
	5	1.68	0.03
	10	3.36	0.06

TABLE IV-4

being examined. The fourth column gives the savings which could result by knowing the increase or decrease in advance and applying the algorithm to the adjusted data. This column is expressed as a percent of total cost.

As can be seen from the table a 10 percent decrease in demand from what a system was designed for would result in a drop in costs of 7.07 percent. However, if it were known at design time that demand would be 10 percent low, the system could be optimally redesigned for that demand level and an additional cost savings of 1.17 percent would result.

Note from Table IV-4 that only the uncertainty in demand has a significant effect on the cost difference between using the adjusted data in the system generated by the nominal data and the system generated by the adjusted data (column 4). However, nearly all changes affect the total cost (Column 3).

The conclusion is that an optimum system and allocations obtained from nominal data will remain very close to optimum under variations in costs. However, variations in demand have a small (but significant) effect on optimum total cost.

V. CONCLUSIONS

As shown in the previous chapter, the algorithm can effectively handle moderate size problems. Computer memory requirements are not a limiting factor to problem size, as they would be with a zero-one algorithm. Computing time required, as seen in Table IV-1, is considerable; however it does not prohibit analysis of large scale problems.

The chief consumer of computing time in the algorithm is the generation of all feasible solutions to the first period problem which cost less than the upper bound for the first period. If this problem could be solved, the algorithm could handle large problems efficiently. If a better initial solution to the problem were available, a lower upper bound for the first period would eliminate many solutions from consideration, yet still maintain optimality.

An important factor toward computing time is the percentage increase of demand and/or costs from period to period (see Table IV-2). Chapter VI suggests a method to avoid the problem. Another important factor is the number of increments of capacity at a warehouse site (Table IV-3).

The algorithm was found to be quite insensitive to changes in costs; that is, if the cost changed from the costs that were used to design the distribution system, the system would be very close to optimum for the new costs. The algorithm was found to be sensitive to changes in demand. Table IV-4 gives the sensitivity results.

Some branch and bound algorithms can be terminated when the first trial optimum solution is reached with only a small loss in optimality and a large reduction in computer time. This algorithm's initial solution, reached after only a few seconds, is generally not close to optimum and the first trial optimum solution occurs near the end of the running time, which makes an early termination unwise.

VI. AREAS FOR FURTHER STUDY

The algorithm uses the linear programming transportation model to solve the allocation problem at each node of the search tree. The transportation model does not necessarily establish the relationship between warehouses and demand centers in which a warehouse is the sole source of supply for a demand center. If it is desired that such a relationship should exist there are two alternatives. The first and most straight-forward is to solve the allocation problem with a zero-one algorithm. The second is to imbed the transportation problem into a branch and bound procedure to eliminate those demand centers with multiple sources from the transportation matrix and fix their allocations outside of the transportation problem. The branch and bound search tree indicates which demand centers to fix and to which warehouses to fix them based on the outcome of the transportation problem. The later approach is much more difficult to implement but should use less computing time especially on large problems.

When fixed or minimum percentage increases in costs and/or demand from period to period are used, the method of calculating the upper bound for each period can be modified so as to significantly reduce the number of nodes that need to be considered and still maintain optimality. A relationship between the optimum cost and the upper-bound in each period would need to be developed and included in the algorithm.

The method of calculating the initial solution to the problem could also be studied. Various alternate ways of getting the initial value of Z^* could be evaluated with the goal of finding an initial solution closest in cost to the final optimum solution.

The algorithm could also be adapted to add on to existing distribution systems. The present first period computations could be eliminated and the first period handled like the other periods. The first period would expand on the existing system. Such a change would simplify the algorithm considerably.

BIBLIOGRAPHY

1. Anastation, A. J. and Weingartner, B. A., Private Communication (with permission).

2. Baumol, W. J., and Wolfe, P., "A Warehouse Location Problem," Operations Research, Vol. 6 (March-April 1958), pp. 252-263.

3. Cooper, L., "Location-Allocation Problems," Operations Research, Vol. 11 (May-June 1963), pp. 331-343.

4. Dachel, A. J., "Locating Warehouses in a Multi-Product Environment," Master's Thesis, Lehigh University, May 1970.

5. Effroymson, M. A., and Ray, T. L., "A Branch-Bound Algorithm for Plant Location," Operations Research, Vol. 14 (May-June 1966), pp. 361-368.

6. Ellwein, L. B. and Gray, P., "Solving Fixed Charge Location-Allocation Problems with Capacity and Configuration Constraints," AIIE Transactions, Vol. 3 (December 1971), pp. 290-298.

7. Feldman, E., Lehrer, F. A., and Ray, T. L., "Warehouse Location Under Continuous Economies of Scales," Management Science, Vol. 12 (May 1966), pp. 670-684.

8. Hadley, G., Linear Programming, Reading, Mass.: Addison-Wesley Co., 1962.

9. Kuehn, A. A., and Hamburger, M. J., "A Heuristic Program for Locating Warehouses," Management Science, Vol. 9 (July 1963), pp. 643-666.

10. Little, J. D. C., Murtz, K. G., Sweeney, D. W., and Karel, C., "An Algorithm for the Traveling Salesman Problem," Operations Research, Vol. 11 (Nov.-Dec. 1963), pp. 972-989.

11. Marks, D. H., "Facility Location and Routing Models in Solid Waste Collection Systems," Ph.D. Thesis, The Johns Hopkins University, November 1969.

12. Revelle, C., Marks, D., and Liebman, J. C., "An Analysis of Private and Public Sector Location Models," Management Science, Vol. 16 (July 1970), pp. 692-707.

13. Sa, G., "Branch-and-Bound and Approximate Solutions to the Capacitated Plant-Location Problem," Operations Research, Vol. 17 (Nov-Dec. 1969), pp. 1005-1016.

14. Spielburg, K., "Algorithms for the Simple Plant Location Problem with Some Side Conditions," Operations Research, Vol. 17 (Jan.-Feb. 1969), p. 85-111.

APPENDIX A

Zero-One Integer Linear Programming Formulation of the Problem

Define the following:

b_{ijt} is the per unit shipping cost from factory i to a warehouse at location j for period t.

h_{jlt} is the per unit handling charge at a warehouse of size increment l at location j for period t.

s_{jkt} is the per unit shipping cost from a warehouse at location j to demand center k during period t.

d_{kt} is the number of units required at demand center k during period t.

g_{jlt} is the fixed cost of maintaining a warehouse of size increment l at location j during period t.

x_{ijlkt} is a zero-one variable indicating whether or not one or more units is shipped from factory i to a warehouse of size l at location j to demand center k during period t.

y_{jlt} is a zero one variable indicating whether or not a warehouse of size increment l is open at location j during period t.

The objective is to minimize

$$\sum_i \sum_j \sum_l \sum_k \sum_t (b_{ijt} + h_{jlt} + s_{jkt})\, d_{kt} x_{ijlkt} + \sum_j \sum_l \sum_t g_{jlt}\, y_{jlt}$$

subject to the following constraints:

1. $x_{ijlkt} = 0,1$ for all i, j, l, k, and t

2. $y_{jlt} = 0,1$ for all j, l, and t

3. Warehouses open must incur the fixed charge.

$$y_{jlt} = \sum_{i} \sum_{k} x_{ijlkt} \quad \text{for all } j,\ l, \text{ and } t$$

4. Once a warehouse is open it cannot be closed.

$$y_{jlt} \geq \sum_{i=1}^{l} y_{j,i,t-1} \quad \text{for all } j,\ l, \text{ and } t$$

5. Only one capacity increment may be used.

$$\sum_{l} y_{jlt} \leq 1 \quad \text{for all } j \text{ and } t$$

6. Factory capacity must not be exceeded.

$$\sum_{j} \sum_{l} \sum_{k} d_{kt}\, x_{ijlkt} \leq f_{it} \quad \text{for all } i \text{ and } t$$

7. Warehouse capacity must not be exceeded.

$$\sum_{i} \sum_{k} d_{kt}\, x_{ijlkt} \leq w_{jlt} \quad \text{for all } j,\ l, \text{ and } t$$

8. Only one factory may serve a warehouse

$$\sum_{i} x_{ijlkt} \leq 1 \quad \text{for all } j,\ l,\ k, \text{ and } t$$

9. Only one warehouse may serve a demand center.

$$\sum_{j}\sum_{l} x_{ijlkt} \leq 1 \quad \text{for all } i,\ k, \text{ and } t$$

10. Demand center must be served.

$$\sum_{i}\sum_{j}\sum_{l} x_{ijlkt} = 1 \quad \text{for all } k \text{ and } t$$

APPENDIX B

USE OF LINEAR PROGRAMMING ON THE TWO-STAGE TRANSPORTATION PROBLEM

In the algorithm to be presented it is necessary to minimize the transportation costs from factory to warehouse to demand center by choosing the sources of supply for each demand center and the sources of supply for each warehouse.

The costs that need to be considered are:

1. The shipping cost from each factory to each warehouse.
2. The cost of handling product at each warehouse.
3. The shipping cost from each warehouse to each demand center.

The constraints that must be met are:

1. Each factory has a maximum capacity.
2. Each warehouse has a maximum handling capacity.
3. Demand at each demand center must be met.

The above costs and restrictions are symbolically represented by the following:

b_{ij} - the per unit shipping cost from factory i to warehouse j.

h_j - the per unit handling cost at warehouse j.

s_{jk} - the per unit shipping cost from warehouse j to demand center k.

f_i - the capacity of factory i.

w_j - the handling capacity of warehouse j.

d_k - the requirements of demand center k.

y_{ijk} - the amount of product shipped from factory i through warehouse j to demand center k.

L - the number of factories.

M - the number of warehouse sites being considered.

N - the number of demand centers.

The objective function is then to minimize:

$$\sum_{i=1}^{L} \sum_{j=1}^{M} \sum_{k=1}^{N} (b_{ij} + h_j + s_{jk})\, y_{ijk}$$

subject to;

$$\sum_{i=1}^{L} \sum_{j=1}^{M} y_{ijk} = d_k \quad , \quad k = 1, 2, ---, N$$

$$\sum_{i=1}^{L} \sum_{l=1}^{N} y_{ijk} \leq w_j \quad , \quad j = 1, 2, ---, M$$

$$\sum_{j=1}^{M} \sum_{k=1}^{N} y_{ijk} \leq f_i \quad , \quad i = 1, 2, ---, L$$

This formulation can be reformulated as an ordinary transportation problem by using the idea of the transshipment problem (see Hadley [8], page 368 ff.). Fictitious stockpiles are established at points which are both origins and destinations. In this case warehouses act as both origins for demand centers and destinations for factories so that the transshipment concept can be applied.

In the transportation formulation each warehouse appears as both an origin and a destination with a large cost (usually referred to as M, but referred to here as A because of conflicting notation) assigned to shipments between warehouses and a zero cost assigned for shipments from each warehouse to itself. If total factory capacity exceeds total warehouse capacity dummy warehouses are introduced as fictitious shipments from one or more warehouses to themselves. If total warehouse capacity exceeds total demand a dummy demand center must be introduced.

The new transportation-compatable formulation is:

minimize
$$\sum_{i=1}^{L+M} \sum_{j=1}^{M+N+1} c_{ij} x_{ij}$$

subject to:

$$\sum_{i=1}^{L+M} x_{ij} = \beta_j \quad , \qquad j = 1, 2, ---, M + N + 1$$

$$\sum_{j=1}^{M+N+1} x_{ij} = \alpha_i \quad , \quad i = 1, 2, ---, L + M$$

where:

$$c_{ij} = b_{ij} + h_j \qquad , \quad i=1, 2, ---, L$$
$$j=1, 2, ---, M$$

$$c_{i+L,\ j+M} = s_{ij}, \qquad i=1, 2, ---, M$$
$$j=1, 2, ---, N$$

$c_{i,j+M} = A,$ $\quad i=1, 2, ---, L$

$\quad j=1, 2, ---, N$

$c_{i+L,j} = 0 \quad ; \quad i=j; \; i=1, 2, ---, M; \; j=1, 2, ---, M$

$c_{i+L,j} = A \quad ; \quad i \neq j; \; i=1, 2, ---, M; \; j=1, 2, ---, M$

$c_{i,M+N+1} = 0, \quad i=1, 2, ---, L$

$c_{i+L,M+N+1} = A \quad i=1, 2, ---, L$

$\alpha_i = f_i, \quad i=1, 2, ---, L$

$\alpha_{i+L} = w_i, \quad i=1, 2, ---, M$

$\beta_i = w_i \quad , \quad i=1, 2, ---, M$

$\beta_{i+M} = d_i, \quad i=1, 2, ---, N$

$$\beta_{M+N+1} = \sum_{i=1}^{L} f_i - \sum_{i=1}^{N} d_i$$

The transportation matrix for the above problem is shown in Figure B-1. Note that, in general, the y_{ijk} variables cannot be obtained in terms of the x_{ij} since product shipped from a given factory to a given warehouse is not identifiable with a particular demand center. This is not a significant drawback since, in practice, such identification takes place at the warehouse.

Thus, the two stage transportation problem can be solved by using the linear programming transportation model.

TRANSPORTATION MATRIX

Org / Des	Whse 1	Whse 2	— — —	Whse M	DC 1	DC 2	— — —	DC N	Slack	Supply
Fcty 1	$b_{11}+h_1$ x_{11}	$b_{12}+h_2$ x_{12}	— — —	$b_{1M}+h_M$ x_{1M}	A $x_{1,M+1}$	A $x_{1,M+2}$	— — —	A $x_{1,M+N}$	0 $x_{1,M+N+1}$	f_1
Fcty 2	$b_{21}+h_1$ x_{21}	$b_{22}+h_2$ x_{22}	— — —	$b_{2M}+h_M$ x_{2M}	A $x_{2,M+1}$	A $x_{2,M+2}$	— — —	A $x_{2,M+N}$	0 $x_{2,M+N+1}$	f_2
⋮	⋮	⋮		⋮	⋮	⋮		⋮	⋮	⋮
Fcty L	$b_{L1}+h_1$ x_{L1}	$b_{L2}+h_2$ x_{L2}	— — —	$b_{LM}+h_M$ x_{LM}	A $x_{L,M+1}$	A $x_{L,M+2}$	— — —	A $x_{L,M+N}$	0 $x_{L,M+N+1}$	f_L
Whse 1	0 $x_{L+1,1}$	A $x_{L+1,2}$	— — —	A $x_{L+1,M}$	s_{11} $x_{L+1,M+1}$	s_{12} $x_{L+1,M+2}$	— — —	s_{1N} $x_{L+1,M+N}$	A $x_{L+1,M+N+1}$	w_1
Whse 2	A $x_{L+2,1}$	0 $x_{L+2,2}$	— — —	A $x_{L+2,M}$	s_{21} $x_{L+2,M+1}$	s_{22} $x_{L+2,M+2}$	— — —	s_{2N} $x_{L+2,M+N}$	A $x_{L+2,M+N+1}$	w_2
⋮	⋮	⋮		⋮	⋮	⋮		⋮	⋮	⋮
Whse M	A $x_{L+M,1}$	A $x_{L+M,2}$	— — —	0 $x_{L+M,M}$	s_{M1} $x_{L+M,M+1}$	s_{M2} $x_{L+M,M+2}$	— — —	s_{MN} $x_{L+M,M+N}$	A $x_{L+M,M+N+1}$	w_M
Demand	w_1	w_2	— — —	w_M	d_1	d_2	— — —	d_N	$\Sigma f_i - \Sigma d_i$	

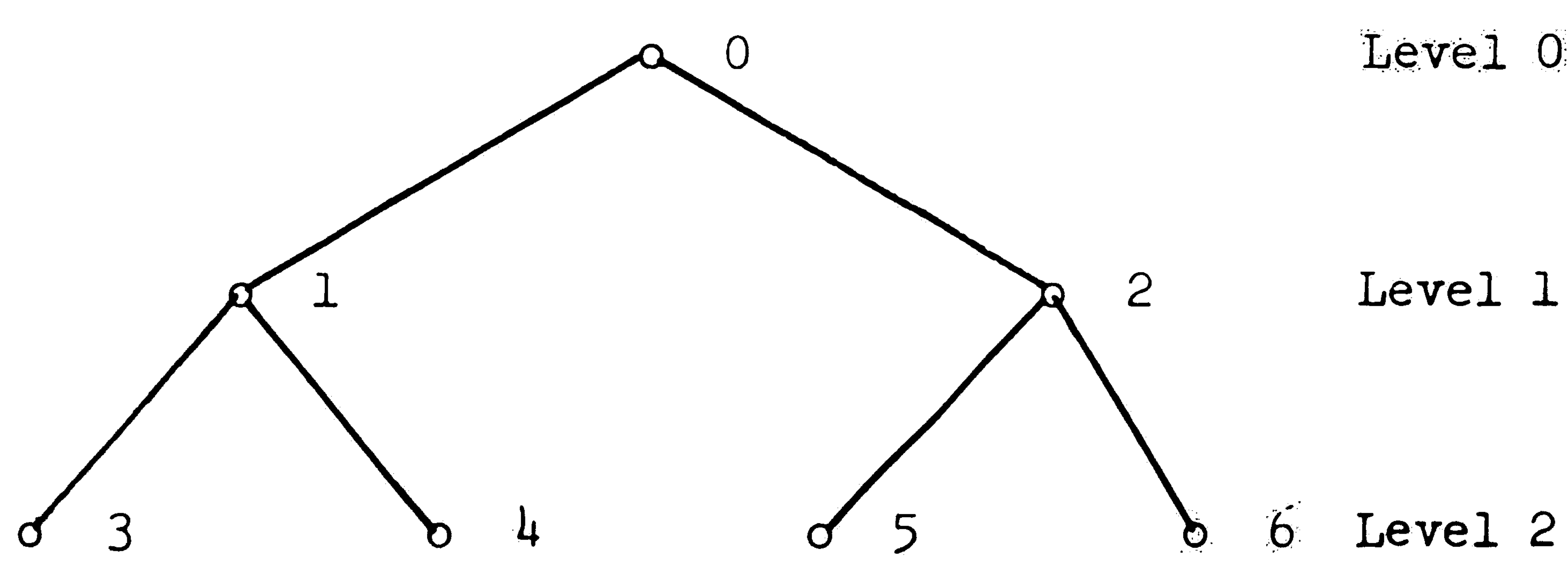
Figure B-1

APPENDIX C

THE BRANCH AND BOUND TECHNIQUE

The branch and bound algorithm was originally developed by Little, et. al.[10] to solve the traveling salesman problem. Branch and bound is basically a means of evaluating all possible solutions to a problem by excluding (hopefully most) solutions which could not possibly be optimum.

The progress toward optimization is followed through the use of a search tree. A typical search tree is shown below.



Each branch indicates the fixing of a variable or variables in the problem at a certain level, with the variable to remain at that level for all branches emanating from the branch. The node at the end of a branch indicates a problem with fewer free variables than the node at the beginning of a branch.

Branch and bound problems are solved by generating branches and nodes which fix the values of variables. Branching continues until all variables have been fixed resulting in a value of an objective function. Once an objective function value has been found upper bounds (for minimization) can be found for the objective function value at each node. Subsequent branching would halt at a node if the minimum possible objective function value at that node were greater than the upper bound for that node.

When it is no longer possible to proceed down the search tree from a node, one must retreat to the node at the next higher level which led to the present node and take another branch down the search tree. When all branches from the source node have been taken the algorithm is completed.

Upper bounds are calculated in such a manner as not to halt progress down a branch which could lead to an optimum solution. Decisions as to which branch to take when moving down the search tree are made by applying an arbitrary rule which attempts to steer the search toward the best solution.

The algorithm in Chapter III uses branch and bound as the method for searching for an optimum solution. The branches in its search tree represent the fixing of warehouse configurations. The levels of the search tree represent the time periods under consideration. Branches from level zero to level one are all the possible combinations of warehouses for period one (many of which are not feasible). Branches from a feasible solution in period one to period two are modifications to the warehouse configurations in period one.

When the final period is reached a solution is checked for optimality. If it is better than the currently best solution (if any) upper bounds can be set for the search tree. Chapter III contains the method for setting upper bounds. The upper bounds for all nodes on a given level are the same.

As each node of the search tree is reached the solution up to that point is checked against the upper bound for that level and the previous level is returned to if the bound is exceeded.

VITA

PERSONAL HISTORY

Name:	William A. Engel, Jr.
Date of Birth:	March 8, 1944
Place of Birth:	Mt. Clemens, Michigan
Parents:	William and Kathleen Engel
Wife:	Patrice E.
Children:	Cynthia D., Christine H.

EDUCATIONAL BACKGROUND

Harbor Beach High School Harbor Beach, Michigan	Graduated - 1962
Michigan State University East Lansing, Michigan Bachelor of Science in Electrical Engineering	Graduated - 1966
Lehigh University Bethlehem, Pennsylvania Candidate for Master of Science in Industrial Engineering	1970-1972

HONORS

Alpha Pi Mu

PROFESSIONAL EXPERIENCE

Western Electric Company, Inc.
White Sands Missile Range, New Mexico
Engineer - Radar Operation and Maintenance
July 1966 - October 1967

Western Electric Company, Inc.
Prince Albert Radar Laboratory, Prince Albert, Saskatchewan, Canada
Planning Engineer - Radar Operation and Maintenance
November 1967 - November 1968

Western Electric Company, Inc.
Assigned to Bell Telephone Laboratories, Inc.
Whippany, New Jersey
Planning Engineer - Radar Test Planning
December 1968 - June 1970

Western Electric Company, Inc.
Princeton, New Jersey
Development Engineer - Lehigh Master's Program
July 1970 - Present